

Creating a Flash driven Halo Menu

When developing websites, one critical issue to decide is what navigational menu your site will use. Developers need to decide during this time how the menu will be integrated into the proposed layout, such as a horizontal or vertical menu. There are many choices from which to decide such as rollovers which use JavaScript, list menu's which are styled using CSS, as well as menu creation tools which automate many facets of this process and are easily customized.

Often times, one doesn't think of using Flash to create a navigation system, however in some situations, it provides another interactive option which is often beyond the capability of HTML and CSS. In this tutorial, we will learn how to create a halo menu, how to show or hide the appropriate submenu and accessibility issues involved with this type of menu.

The finished page

http://midwestwebdesign.net/tutorials/flash/halomenu/halo_menu_finished.html

Download Project Files

<http://midwestwebdesign.net/tutorials/flash/halomenu/halomenu.zip>

Advantages/Disadvantages to the menus mentioned

When you sit back and think about the menu choices you have, no menu presented today is perfect. For example, if you're using a menu system with CSS and JavaScript, the following can be problematic:

- If JavaScript is disabled in a visitor's browser, the menu will not work
- If the CSS isn't developed with cross compatibility in mind, its behavior is likely to be inconsistent among different browsers and operating systems.

When using Flash to create a menu, the following can also be problematic:

- If the visitor doesn't have the correct version of Flash player installed, the navigation will not render or work properly in the browser.
- You lose the ability to interact with assistive readers. When using menu systems which use regular HTML anchor tags in conjunction with CSS, you can provide assistive readers alternative ways of navigating your website such as tabbed key support.

With this said, it's critically important developers keep in mind the intended audience of the website. Once all factors have been accounted for, choose the navigation system which best suits your needs.

Create the file

Inside Flash, create a new file named **halomenu.fla** and create the following hierarchy:

ActionScript	
	ActionScript
	Stops
	Labels
Main Header	
	page header
Top Buttons	
	home
	services
	gen info
	tutorials
Services Sub Menu	
	main menu
Gen Info Sub Menu	
	main menu
Tutorials	
	main menu

In the above diagram, folders are on the left and layers are on the right. Feel free to assign your folders and layers the colors you want.

The page header

We will create our page header first, since it's the easiest. In the page header layer follow these steps:

- Right click on key frame one and select **Insert Key Frame**
- Select the **text tool button** ("A") and drag a text field instance on the stage
- Inside the text field type **Halo Menu** using the following properties:
 -
- Press **F8** and in the symbols window, type < > and select **graphic**. Continuing, follow these steps to create the fade sequence which will be explained later:
 - Select frame one, in the properties panel, select **Alpha** as color
 - Right click frame ten and select **Insert Key Frame**
 - With frame ten selected, in the properties panel, select **none** as color

Main menu buttons

We have four main menu buttons which make up the central navigation. The only difference between these buttons is the x-y coordinates. As a result, we will use the home button as our example during the main menu button creation process. In the home layer as shown below:

Follow these steps:

- Right click key frame one and select **Insert Blank Key Frame**
- Select the **text tool button (“A”)** and drag a text field instance on the stage
- Inside the text field, type **home**
- Make sure the text field is static. In the properties panel, provide the following for font:
 - Font family: Arial
 - Size: 20
 - Color: Dark blue (#14314F)
- With the text selected, press **F8** and in the symbol window type **home** and then select **button**.

With the object selected, follow these steps:

- In the properties panel, type **homeBtn** in the Instance text box
- Double click the button inside the library to enter symbol editing mode
- Rename your layer to **home**.
- Create the other three states for your button, most importantly the hit state to make your button clickable. Once completed, exit symbol editing mode by clicking the blue background arrow:

Position the button at the following x-y coordinates:

- X : 58
- Y : 86

Repeat the above button creation process for services, general information and tutorials, replacing instance names in the library and on the stage in the Instance text box. Once completed, change the x-y coordinates for each button as follows:

Services

- X : 154
- Y : 86

General Information

- X : 262
- Y : 86

Tutorials

- X : 402
- Y : 86

Fade Sequence

When the movie loads, we want the page header as well as our four main menu buttons to fade in until frame ten. While we have completed this for the page header, we need to complete the process for our buttons. Since the process is the same for all buttons, we will use home as our example. In the home layer, follow these steps:

- Right click frame ten
- Select **Insert Key Frame**.
- In the properties panel, select **none** as color

We will create a motion tween for each button to fade in smoothly as follows:

- Right click in the empty frames area as shown below:
- Right click and select **Create Motion Tween**.

Finally, in the **Stops** layer, complete the following:

- Right click frame ten
- Select **Insert Key Frame**
- Press **F9** on your keyboard to open the actions panel and insert the following code:

```
stop();
```

Currently, when the movie loads, it will begin at frame one and play to frame ten and loop continuously. To prevent this we add a stop command at the end of the frame sequence which is frame ten. Save your file and refer to < > if you need to compare.

http://midwestwebdesign.net/tutorials/flash/halomenu/halomenu_step_one.html

Sub menu buttons

Since each of the sub menu buttons are the same except for positioning placement on the stage, we will use the services sub menu for the creation process. First, we will create an

empty movie clip to hold our sub menu buttons. On the layer named **main menu** under services follow these steps to create the movie clip:

- Right click frame ten
- Select **Insert Blank Key Frame**
- Press **F8**, and in the symbol window type **servicessubmenu**.

Exit symbol editing mode by clicking the blue backward arrow button as shown below:

Return to the submenu movie clip and make sure you are on frame ten. From the library, drag the movie clip on the stage at the following coordinates:

- X : 196
- Y : 152

With the movie clip selected, in the properties panel, in the Instance text field, type **servicesmc**. With the movie clip selected, double click to enter symbol editing mode. Inside symbol editing mode for the movie clip, create a hierarchy as follows:

Action Script	
	Action Script
	Stops
	Labels
Services Navigation	
	consulting
	freelance
	hosting

Since each of our sub menu buttons are only different in position placement, we will use consulting as our example. Follow these steps to create the button:

- In the consulting layer for frame two
 - Right click and select **Insert Key Frame**
 - Left click the **text tool** (“A”) and drag an instance of text on the stage
 - Type **consulting**
- Assign the following for font:
 - Font: Arial
 - Size: 20
 - Color: Dark red (#990000)

With the text field selected:

- Press **F8**
- In the symbol window, type **consulting** and select **button**

Make sure you create the other three states of the button, most importantly, the hit state to make your button clickable. Exit symbol editing mode by clicking the blue backward arrow button as shown below:

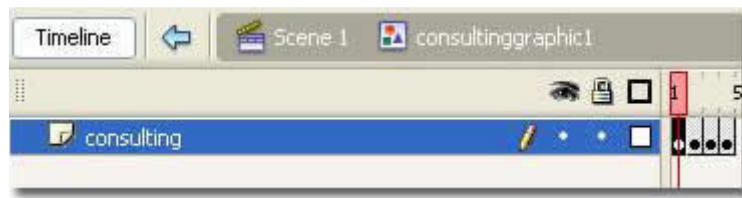
You should return to the services movie clip. Inside the library complete the following:

- Right click the consulting button
- Select **duplicate**

In the symbol window:

- Type **consultinggraphic**
- Select **graphic**
- Press **OK**

Inside the library, double click the graphic and you will see this:



You will notice we have four key frames. The reason for this is simple, when we converted our button to a graphic, Flash retained the four states to our button. We don't need these extra frames so delete frames two through four. Continuing, exit symbol editing mode by clicking the blue backward arrow button as shown below:

You should return to the services movie clip.

Fade Sequence – Submenu buttons

When a visitor moves their mouse over one of the main menu buttons, we want to fade the sub menu buttons in. Follow these steps to create the fade sequence for the buttons:

- Inside the services movie clip, select key frame two in the consulting layer
- In the properties panel, select **Alpha** as color

For key frame six, complete the following:

- Select key frame six in the consulting layer
- Right click and select **Insert Key Frame**
- In the properties panel, select **none** as color

Fade Out Sequence – Submenu buttons

When a visitor moves their mouse over a main menu button, the appropriate submenu will fade in below. When the visitor subsequently rolls out and over another main menu button, we need to fade out the submenu which is currently showing and fade in another submenu. Since each of the services submenu buttons has identical properties, we will create the fade out sequence using the consulting button as our example. Follow these steps to create the sequence:

- Select key frame seven
- Right click and select **Insert Blank Key Frame**
- Left click the consulting graphic from the library and place in the same x-y coordinate as the button

For key frame eleven, complete the following:

- Select key frame eleven
- Right click and select **Insert Key Frame**
- Set color to **Alpha**

For key frame seven:

- Set color as **none**

To prevent the fade out sequence from looping continuously, follow these steps:

- Inside the services movie clip, select the **stops** layer
- Select key frame one
- Right click and select **Insert Key Frame**

Press F9 to open the actions panel and type the following:

```
stop();
```

Also add a stop command to key frame six and eleven. To change the opacity (fade in/out), follow these steps:

- Select key frame seven
- Right click and select **Insert Blank Key Frame**
- Left click the consulting graphic from the library and place in the same x-y coordinate as the button
- Select key frame eleven
- Right click and select **Insert Key Frame**

For key frame seven:

- Set color to **none**

For key frame eleven:

- Set color to **Alpha**

Repeat these steps for the freelance and hosting buttons.

Showing and Hiding Sub Menus

In order for the services submenu to show or hide, we need to set two frame labels, one which will be set to show, and the other set to hide. The reason we use frame labels is in case we move the frames which have content to another frame, it won't break our code. For example, if in action script we instruct the submenu to show by using a frame number:

```
servicesBtn.onRollOver=function(){
    servicesmc.gotoAndPlay("10");
}
```

If the show sequence is moved to another frame number such as 20 our code will not work. Using frame labels solves this issue. To add frame labels to our submenu, make sure you are inside the services submenu, and select key frame two in the labels layer and follow these steps:

- Right click and select **Insert Key Frame**
- In the properties panel, type **show**

For key frame seven:

- Right click and select **Insert Key Frame**
- In the properties panel, type **hide**

Choosing which submenu to show

To determine which menu we want to show, we need to assign a value to a variable. In this case, since we are dealing with the services submenu, we set the variable, **menuOpen** to services as shown below:

```
_parent.menuOpen="services";
```

Exit the services movie clip by clicking the blue backward arrow button as shown below:

You will return to the main stage. For the general information and tutorials submenu, you need to repeat the same process we went through for the services submenu. The only difference is the x-y coordinates for general information and tutorials movie clips which are as follows:

General Information

- X : 330.0
- Y : 152.0

Tutorials

- X : 448.0
- Y : 152.0

Continuing, make sure on the action script layer, in frame two, you change the value of the variable (menuOpen) inside the general information and tutorials movie clips to the following:

```
_parent.menuOpen="geninfo";  
_parent.menuOpen="tutorials";
```

Save your file and use < > as a comparison.

Why did we convert our sub menu buttons to graphics?

Though the reasoning requires some explanation, the answer is scope. When you hide the visibility of the button on the stage, it's still active. In other words, when the visitor hovers their mouse over one of the main menu buttons, for example, the services button, the corresponding submenu shows. At this point, the submenu movie clip is in scope to the rest of the movie. Therefore, when the visitor hovers over another main menu button, for example tutorials, the services submenu hides visually. At this point, if the visitor hovers over the location on the stage where the services submenu resides, the mouse pointer would change to a hand, which would lead to an awkward behavior for the visitor.

To prevent this from happening, we use a graphic symbol. By duplicating the button, we acquire the same font family and size, and by deleting frames two through four we eliminate the other states associated with a button. Continuing, by placing the graphic symbol in the hide sequence instead of the button, when the sequence is played, it creates the illusion the button is removed from the stage. Even though the graphic symbol is still in scope, that is, still on the stage, graphic symbols don't exhibit button behavior, thus this technique eliminates any potential behavior which might confuse a visitor using your menu on your website.

Why did we start our sub menu buttons on key frame 2 and stop our movie clips on key frame 1?

The reason for this is two fold. First, we don't want our submenu buttons in scope or to show until the visitor moves their mouse over the main menu buttons. Right now, without a stop command, when the movie loads, all three submenu would show once the play head in the main timeline hit frame ten. Therefore, we place a stop command on key frame one inside each movie clip. As a result of the stop command being placed on key frame one, we can't instruct the submenu movie clips through action script to play in the same key frame. You might be asking why? Because there would be conflicting commands taking place, that is, you can't instruct the play head inside the movie clip to play and stop in the same key frame. As a result, we move the submenu buttons to key frame two.

The functionality

Since we have completed the user interface, let's review briefly the intended behavior of the menu. When the movie loads, we will see the page header and four main menu buttons fade in. As we move our mouse over each of the main menu buttons, the appropriate submenu will show. Furthermore, as we move our mouse over the services submenu and subsequently mouse over tutorials, we will see the services submenu hide and the tutorials menu show. Continuing, if a submenu is visible and the visitor moves their mouse over the home button, the submenu which is currently visible will be hidden. In other words, a simple toggling behavior accomplished by conditional logic.

Sub menus showing

First, we will add action script to show each submenu. From the main timeline, select the action script layer in frame ten and follow these steps:

- Right click and select **Insert Key Frame**
- Press **F9** to open the actions panel and type the following:

```
servicesBtn.onRollOver=function(){
    servicesmc.gotoAndPlay("show");
}
geninfoBtn.onRollOver=function(){
    geninfomc.gotoAndPlay("show");
}
tutorialsBtn.onRollOver=function(){
    tutorialsmc.gotoAndPlay("show");
}
```

We use the **onRollOver** method of the button, specify which movie clip, such as services submenu and use the **gotoAndPlay** method of the movie clip, followed by a parameter, in

this case, show. Recall, show is the name of the frame label associated with playing the show sequence which shows the submenus. Save your file and use <> as a comparison.

Adding the hide functionality

To hide our submenus when another main menu button is hovered over, we add the following code:

```
var menuOpen:String="close";
function isitOpen(){
    if(menuOpen=="services"){
        menuOpen="close";
        servicesmc.gotoAndPlay("hide");
    }
    else if(menuOpen=="geninfo"){
        menuOpen="close";
        geninfomc.gotoAndPlay("hide");
    }
    else if(menuOpen=="tutorials"){
        menuOpen="close";
        tutorialsmc.gotoAndPlay("hide");
    }
}
```

First, we create (declare) a variable named **menuOpen** by using the **var** keyword. We set it as a string data type and assign a value of **close**, which will be used later. Next, we create a function named, **isitOpen**. Inside this function, we check the value of our variable, **menuOpen**. We can determine the value of the variable by using a Boolean check, that is, a double equals sign. In the first condition, we check if the value of the **menuOpen** variable is equal to services. If it is, then the condition equates to true and we set the **menuOpen** variable to close and instruct the services submenu to play the hide sequence. However, if the first condition equates to false, then we nest two additional conditional checks to test the value of the **menuOpen** variable for the remaining two submenus.

Remember, inside our submenu movie clips, for frame 2 in the action script layer, we set the value of **menuOpen** to the appropriate sub menu. As a result, when the movie loads and a main menu button are hovered over, such as services, the show sequence is played. Subsequently, **menuOpen** is set to **services**, thus the first if statement equates to true, the variable is then set to **close** and plays the hide sequence. Finally, we add the following action script to the roll over method of the home button to hide any submenu which might be visible:

```
homeBtn.onRollOver=function(){
    isitOpen();
}
```

We simply call the function, **isitOpen** to determine if any submenu is showing. If it is, we hide the submenu. Save your file and preview the results. Hover over any of the main menu buttons, and then hover over home, should hide the appropriate sub menu.

http://midwestwebdesign.net/tutorials/flash/halo_menu_stepfour.html

Completing the menu

Currently, hovering over services will show its submenu. However, hovering over general information or tutorials will show their submenu while the existing services submenu remains visible when it should be hidden. To fix this for each button, we add the following code:

```
servicesBtn.onRollOver=function(){
if(menuOpen=="services"){
    null;
}
else if(menuOpen=="close" || menuOpen=="geninfo" || menuOpen=="tutorials"){
    isitOpen();
    servicesmc.gotoAndPlay("show");
}
}
```

First, we check the value of **menuOpen**. If it's equal to services, we do nothing, since the menu is currently visible. If it's equal to **close**, **geninfo**, or **tutorials**, we first call the function, **isitOpen**. The reason for calling the function first is to hide the sub menu which is currently visible before showing the correct submenu, such as when the visitor moves their mouse over services, then tutorials. You might be asking, how are we hiding the services submenu? Recall, when we created (declared) our variable; we assigned a value of close. If the value of **menuOpen** is equal to services, then we know the submenu is visible and that **menuOpen** is now set to a value of close. We can check the value of one variable for three different values and implement a toggle behavior (show/hide) by using the logical operator **OR** which in action script is this symbol: ||

After the function call, we instruct the appropriate movie clip, in this instance, services, to play the show sequence. You might find the logic here a bit odd, since we are instructing all submenus to be hidden first, and then we instruct the services submenu to show. Our reason is simple; we are ensuring that on a roll over of a main menu button, only the correct submenu will show. For the other two main menu buttons, we alter the last if statement to the following:

```

geninfoBtn.onRollOver=function(){
    if(menuOpen=="geninfo"){
        null;
    }
    else if(menuOpen=="close" || menuOpen=="services" ||
menuOpen=="tutorials"){
        isitOpen();
        geninfomc.gotoAndPlay("show");
    }
}

```

```

tutorialsBtn.onRollOver=function(){
    if(menuOpen=="tutorials"){
        null;
    }
    else if(menuOpen=="close" || menuOpen=="services" ||
menuOpen=="geninfo"){
        isitOpen();
        tutorialsmc.gotoAndPlay("show");
    }
}

```

For both buttons, we switch out the last two checks on the **menuOpen** variable. For example, the general information button the last two checks on **menuOpen** need to be **services** and **tutorials**. For the tutorials button, switch out the last two checks on **menuOpen** to be **services** and **geninfo**. Save your file and preview the results, your menu should be fully functional.

http://midwestwebdesign.net/tutorials/flash/halomenu/halo_menu_stepfive.html

Conclusion

In this article you learned to create a version of the halo menu. More importantly, you learned how to:

- Type of menus available
- Advantages and disadvantages for each type of menu
- Create button symbols using the text tool
- Create submenus using movie clips
- Duplicate buttons and convert them to graphics to solve scope issues
- Create a variable, function and assign different values to the variable
- Create simple conditional logic using if statements to determine the value of the variable and perform an appropriate action.
- How to check a variable for three different values using the logical operator OR.

You can take the knowledge gained here and customize your halo menu to your imagination!

If you have questions, please follow the link below.

<http://midwestwebdesign.net/tutorials/contact.aspx>